MMME 3085 Jan 23 Example Exam Solutions

Question 1:

```c
# include <stdio.h>
# include <stdlib.h>
# include <math.h>

int main()
{
    float num;
    float tol = 0.0;  // Specify tolerance for iterative process
    float root = 0;
    float nextRoot = 100;  // Initialise to a value which will execute loop at least
once
    int iterations = 0;

    printf("Input a value to calculate the square root: ");
    scanf("%f", &num);

    // Abort program if negative number input
    if ( num < 0.0 )
    {
        printf( "Cannot calculate the square root of a negative number, aborting
program\n");
        exit(0);
    }

    tol = num * 0.00001 * 0.01;  // Convergence when within 0.00001 % of original
number

    // Initial guess at the root
    root = num/2.0;

    // Execute until results are within tolerance
    while ( (fabs(nextRoot - root) > tol))
    {
        root = nextRoot;
        nextRoot = 0.5* (root + num/root);
        iterations++;
    }

    // Output the result
    printf( "The square root of %10.5f is %10.5f\n", num, nextRoot);
    printf("Number of iterations = %d", iterations);
}
```

Question 2:

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int calcPolygonArea( int Sides, float length, float *area);

int main()
{
    int numSides = 0;
    float length = 0.0;
    float polygonArea = 0.0;

    // Input number and length of sides of polygon
    printf( "Input number of sides in polygon: ");
    scanf("%d", &numSides);
    printf( "Input length of polygon side: ");
    scanf("%f", &length);

    // Calculate area of polygon
    if ( calcPolygonArea( numSides, length, &polygonArea) )
        printf( "Area of polygon with %d sides of length %6.2fm is %6.2fm^2\n",
numSides, length, polygonArea);
    else
        printf("Cannot calculate area, invalid input\n");

    return 0;
}

int calcPolygonArea( int Sides, float length, float *area)
{
    // Check for valid input and return if invalid
    if ( Sides < 3 || length <= 0.0)
        return 0;
    // Calculate area of polygon
    *area = (length*length*(float)Sides)/ (4*tan(M_PI/Sides));
    return 1;
}
```

Question 3:

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    FILE *fIn;
    char filename[50];
    int format;
    char c;

    // Input filename and open file
    printf("Enter the name of the input file: ");
    scanf("%s", filename);

    if ( (fIn = fopen(filename, "r")) == NULL)
    {
        printf("Failed to open input file, terminating program\n");
        exit(0);
    }

    // Select output format
    printf( "Enter format for output (1-3): ");
    scanf("%d", &format);

    // Terminate program if incorrect format
    if ( format != 1 && format != 2 && format !=3)
    {
        printf("Incorrect format, terminating program\n");
        fclose(fIn);
        exit(0);
    }

    // Loop until reach EOF
    while ( !feof(fIn) )
    {
        // Read a single character
        c = fgetc(fIn);

        // Output depending on format selected
        switch(format)
        {
            case 1:
                printf("Character read was %c\n", c);
                break;
            case 2:
                printf("Ascii value of character is %d\n", c);
                break;
```

```
            case 3:
                printf("Ascii value of %c is %d\n", c,c );
                break;
            default:
                printf("Couldn't print character\n");
                break;
        }
    }

    // Close the file
    fclose(fIn);
}
```

Question 4:

A.

i) Linear encoder

1. Most likely is a linear encoder (e.g., moire-based grating system). [1 mark]
2. Signal is a quadrature signal. [1 mark]
3. Needs some form of quadrature decoder to interpret the signal. [2 marks]
4. To avoid any interference, use differential signals e.g., produced by a "line driver". [1 mark]

ii) A temperature sensor (thermocouple)

1. The temperature range is such that a thermocouple is needed. [1 mark]
2. Analogue signal in the range of millivoltages. [1 mark]
3. Cold junction compensation and software to interpret voltage as a temperature using the thermocouple tables/polynomials as well as ADC (or use the built in one). [2 marks]
4. A sensitive amplifier is needed. [1 mark]

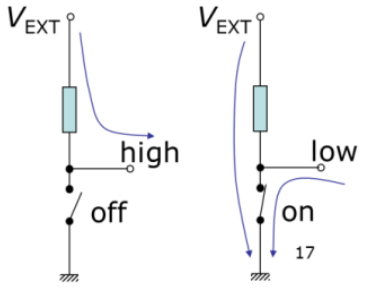
iii) Strain gauge

1. A strain gauge can be used to convert the strain into force. [1 mark]
2. Analogue signal in the range of millivoltages. [1 mark]
3. The strain gauge changes the stain into electrical resistance change. Using a Wheatstone's bridge, it can be then converted into a voltage change. Then an amplifier is used to scale up the signal. An ADC will be required to interface with a microprocessor (or use the built in one). [2 marks]
4. A low-pass filter to eliminate unwanted noise frequencies. [1 mark]

B.

Output is connected to an external voltage supply via a pull-up resistor and is either pulled up to that voltage (if transistor does not conduct) or is shorted to ground (if transistor conducts) as shown in the figure below:

[2 marks]



[2 marks]

Used to obtain logic output voltage not limited to (say) 5V as for a normal (totem pole) output and to work as isolator between the control and power circuits.

[1 mark]

C.

Solution: use three-state (tri-state) buffer, has the states:

- high
- low
- High impedance (i.e., open circuit)

[2 marks]

When enabled:

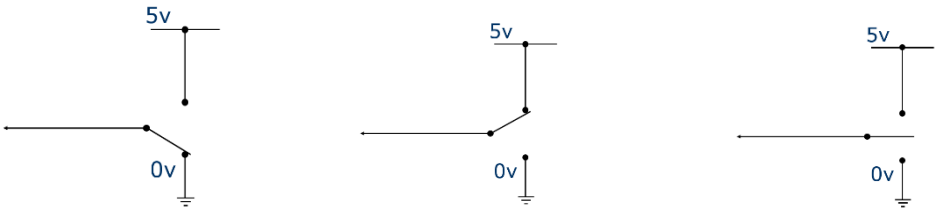
- if input is high, puts high signal onto bus
- if input is low, puts low signal onto bus

When disabled: Connection to bus is broken ("high impedance state") like an open switch

Enable/disable depends upon whether

- The address on the address bus corresponds to the address identifying the interface
- Whether the data at the address is to be read or written (status of read/write line)

[2 marks]



[2 marks]

Question 5:

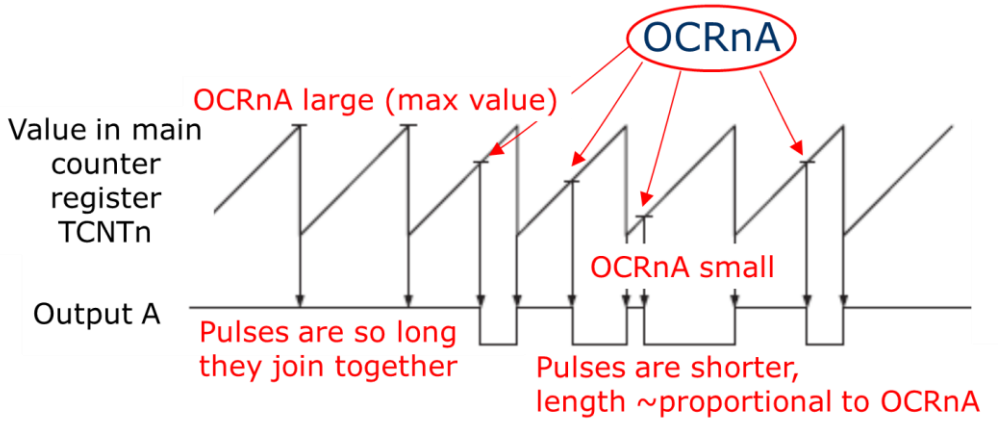
A.

Students are not expected to identify registers but are expected to know the following:

- counter register (e.g., TCNTn) cycles from zero to maximum value then wraps back to zero.
- Output is set to 1 at each cycle start (i.e., TCNTn reaches max) and reset to 0 when a threshold is reached (i.e., TCNTn= a threshold (e.g., the value in OCRn**A)**).

Full marks for conveying concept of how varying value sets and resets output, with diagram (as below) with labels which support concept.

[6 marks]

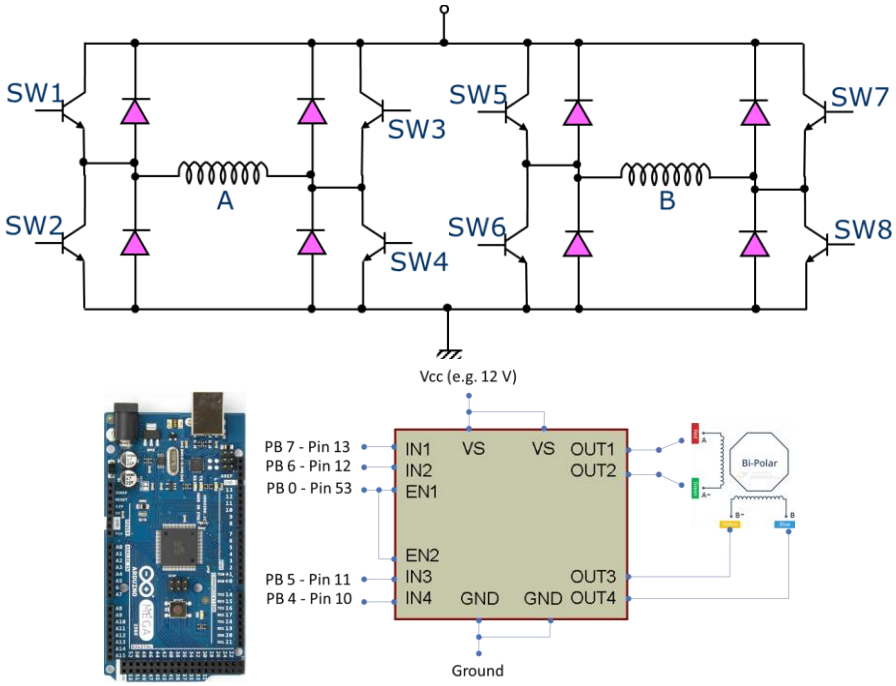


[5 marks]

Note: this is the version of PWM on Arduino taught in class, it is actually "fast PWM".

B.

The connection is as follows:



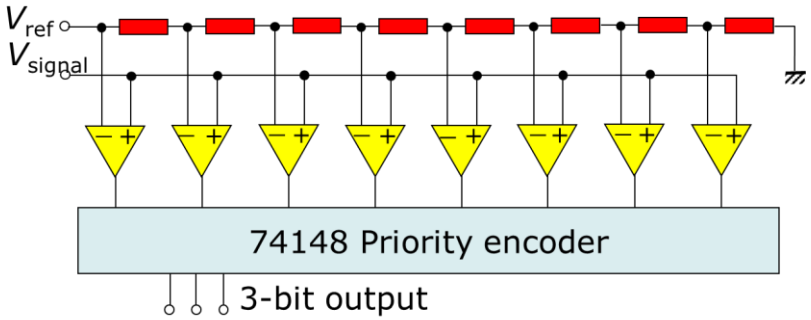
Note: The student is not supposed to identify particular pins (as long as they are digital output).

[6 marks]

C.

- A large number of comparators is used as in the below figure (1 per increment)
- Each one compares input with reference (V_{ref} is divided using the resistors)
- Priority encoder converts the output from the comparators into a binary number

[3 marks]



[4 marks]